



RECEIVED

DEC 15 2004

Technology Center 2600 CUSTOMER NUMBER 25268

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants: Chris Yoochang Chung et al. Attorney Docket No: UNIV0123
Serial No: 10/053,431 Group Art Unit: 2676
Filed: January 17, 2002 Examiner: Nguyen, Hau H.
Title: PROGRAMMABLE 3D GRAPHICS PIPELINE FOR MULTIMEDIA
APPLICATIONS

PETITION FOR WITHDRAWAL OF RESTRICTION REQUIREMENT UNDER 37 CFR 1.144

Bellevue, Washington 98004

December 6, 2004

TO THE DIRECTOR OF THE PATENT AND TRADEMARK OFFICE:

REMARKS

Status of Restriction Requirement

In the final Office Action dated September 3, 2004, the Examiner responded to applicants' election and traverse of the Restriction requirement, making the Restriction final. Applicants hereby petition the Director to withdraw the Restriction requirement for the reasons set forth below. The petition fee is enclosed herewith. A listing of the claims as currently amended in the Response to the Final Office Action that is being filed concurrently herewith is presented below for the convenience of the Director.

The Examiner has restricted the application into four groups as follows:

Group I, Claims 1, and 3-16, which the Examiner asserts are "drawn to 'programmable graphics pipeline' classified in class 345, subclass 506;"

Group II, Claims 17-22, which the Examiner asserts are "drawn to 'processing graphics data and media data' classified in class 345, subclass 546;"

Group III, Claims 23-29, which the Examiner asserts are "drawn to 'processing variable length data' classified in class 345, subclass 522;"

Group IV, Claim 30, which the Examiner asserts is "drawn to 'caching texture data and media data' classified in class 345, subclass 552.

12/10/2004 JBA/LINAN 00000028 10053431

02 FC:1464

130.00 OP

-1-

2735-3496-3889PT
UNIV0123-1-21/0123AM_PETITION.doc

LAW OFFICES OF RONALD M. ANDERSON
600 - 108th Avenue N.E., Suite 507
Bellevue, Washington 98004
Telephone: (425) 688-8816 Fax: (425) 646-6314

1 **Amendment to the Claims**

2 1. (Currently Amended) A programmable graphics pipeline for processing multimedia data
3 comprising:

4 (a) an instruction cache for storing at least graphics and media instructions;
5 (b) a first register file for storing the multimedia data and intermediate data;
6 (c) a first vector functional unit in communication with the instruction cache and
7 the register file, said vector functional unit performing at least graphics and media instructions, to
8 produce at least graphics and media data; and

9 (d) an enhanced texture cache in communication with the first vector functional
10 unit, the first vector functional unit obtaining from said enhanced texture cache a vector of at least
11 one partition of the multimedia data, wherein said enhanced texture cache is not limited to storage of
12 one type of multimedia data.

13 2. (Previously Cancelled)

14 3. (Previously Presented) The programmable graphics pipeline of Claim 1, wherein the
15 enhanced texture cache comprises:

16 (a) a line buffer that provides multiple read ports for accessing the multimedia
17 data; and

18 (b) a cache area in communication with the line buffer, said cache area storing the
19 multimedia data.

20 4. (Original) The programmable graphics pipeline of Claim 3, further comprising an
21 enhanced texture address unit in communication with the enhanced texture cache, said enhanced
22 texture address unit being employed for converting multiple inverse-mapped source coordinates into
23 absolute memory addresses of the multimedia data in an arbitrary-sized block.

24 5. (Original) The programmable graphics pipeline of Claim 4, wherein the enhanced texture
25 address unit further generates filter coefficients for the multiple inverse-mapped source coordinates
26 for one of a bilinear filtering and a trilinear filtering of the data.

27 ///

28 ///

29 ///

30 ///

1 6. (Original) The programmable graphics pipeline of Claim 4, further comprising an
2 enhanced rasterization unit in communication with the enhanced texture address unit, said enhanced
3 rasterization unit being employed for generating:

- 4 (a) a plurality of destination coordinates for at least one of:
5 (i) a plurality of primitives being rendered; and
6 (ii) a media processing output; and
7 (b) a plurality of source coordinates for at least one of:
8 (i) texture data associated with the plurality of destination coordinates for
9 the plurality of primitives; and
10 (ii) media data associated with the plurality of destination coordinates for
11 the media processing.

12 7. (Original) The programmable graphics pipeline of Claim 6, wherein the plurality of
13 primitives include at least one of:

- 14 (a) dots;
15 (b) lines;
16 (c) triangles;
17 (d) rectangles; and
18 (e) polygons.

19 8. (Original) The programmable graphics pipeline of Claim 6, further comprising an
20 enhanced Z-buffer unit in communication with the enhanced rasterization unit and the enhanced
21 texture address unit, said enhanced Z-buffer unit being employed for loading source coordinates of a
22 primitive being rendered to provide the enhanced texture address unit with variable access to the
23 source coordinates if perspective address generation cannot be used.

24 9. (Original) The programmable graphics pipeline of Claim 8, wherein the enhanced
25 Z-buffer unit is further employed for determining a depth of a new source pixel in relation to an old
26 source pixel.

27 10. (Original) The programmable graphics pipeline of Claim 1, further comprising a
28 blending unit that is in communication with the first vector functional unit, said blending unit being
29 employed for:

30 ///

1 (a) combining a graphics data output from the first vector functional unit with a
2 color value to produce a blended value; and

3 (b) combining the blended value with a destination pixel value.

4 11. (Original) The programmable graphics pipeline of Claim 1, further comprising an output
5 buffer in communication with the first vector functional unit, said output buffer being employed for
6 concatenating one of successive image outputs and successive video outputs from the first vector
7 functional unit, to reduce a number of subsequent write transactions to a memory.

8 12. (Previously Presented) The programmable graphics pipeline of Claim 1, further
9 comprising an output buffer in communication with the first vector functional unit and the enhanced
10 texture cache, said output buffer being employed for:

11 (a) concatenating successive media processing output from the first vector
12 functional unit to produce concatenated data;

13 (b) providing the concatenated data as input to the enhanced texture cache; and

14 (c) converting a compressed format of destination coordinates into a plurality of
15 destination addresses for media processing output data.

16 13. (Original) The programmable graphics pipeline of Claim 10, further comprising a write
17 buffer in communication with the blending unit, said write buffer sending multiple write transactions
18 to a memory, to reduce page misses.

19 14. (Original) The programmable graphics pipeline of Claim 11, further comprising a write
20 buffer in communication with the output buffer, said write buffer sending multiple write transactions
21 to a memory, to reduce page misses.

22 15. (Original) The programmable graphics pipeline of Claim 1, further comprising a
23 configuration register in communication with the first vector functional unit, said configuration
24 register providing the first vector functional unit with partitioned data access parameters and a
25 location of a current instruction thereby instructing the vector functional unit to perform one of a
26 graphics process and a media process.

27 16. (Original) The programmable graphics pipeline of Claim 1, further comprising:

28 (a) a second vector functional unit in communication with the instruction cache
29 and performing the graphics and media instructions being performed by the first vector functional
30 unit, to produce parallel graphics and media data; and

1 (b) a second register file in communication with the second vector functional unit,
2 said second register file storing additional multimedia data and additional intermediate data.

3 17. (Previously Presented) A method for producing one of graphics pixel data and media
4 output data, comprising the steps of:

5 (a) obtaining configuration data from a host processor, said configuration data
6 comprising a partitioned data size and a location of an instruction;

7 (b) performing graphics rendering processing on graphics texture data with a
8 programmable graphics rendering pipeline to produce graphics pixel data when the partitioned data
9 size and the instruction correspond to graphics processing; and

10 (c) performing media processing on media source data with the programmable
11 graphics rendering pipeline to produce media output data when the partitioned data size and the
12 instruction correspond to media processing.

13 18. (Previously Presented) The method of Claim 17, wherein the step of performing graphics
14 rendering processing comprises the steps of:

15 (a) producing a destination coordinate;

16 (b) producing a mapped texture coordinate corresponding to the destination
17 coordinate;

18 (c) producing a plurality of surrounding memory addresses in which surrounding
19 texture data are stored, said surrounding texture data corresponding to a plurality of surrounding
20 texture coordinates surrounding the mapped texture coordinate;

21 (d) retrieving the surrounding texture data from the plurality of surrounding
22 memory addresses, and storing the surrounding texture data retrieved in a cache; and

23 (e) performing interpolation on the surrounding texture data in the cache to
24 determine a texture value at the mapped texture coordinate as the graphics pixel data for the
25 destination coordinate.

26 19. (Original) The method of Claim 17, wherein the step of performing media processing
27 corresponds to image processing comprises the steps of:

28 (a) producing a destination coordinate;

29 (b) producing at least one mapped image coordinate corresponding to the
30 destination coordinate;

1 (c) producing at least one memory address where image data are stored
2 corresponding to the at least one mapped image coordinate;

3 (d) retrieving the image data from the at least one memory address, and storing the
4 image data retrieved in a cache; and

5 (e) performing an image manipulation function on the image data in the cache to
6 produce image pixel data at the destination coordinate as the media output data.

7 20. (Original) The method of Claim 17, wherein the step of performing media processing
8 corresponds to video processing comprises the steps of:

9 (a) producing a destination coordinate;

10 (b) producing at least one mapped video coordinate corresponding to the
11 destination coordinate;

12 (c) producing at least one memory address where video data are stored
13 corresponding to the at least one mapped image coordinate;

14 (d) retrieving the video data from the at least one memory address, and storing the
15 video data retrieved in a cache; and

16 (e) performing a video manipulation function on the video data to produce video
17 pixel data at the destination coordinate as the media output data.

18 21. (Previously Presented) The method of Claim 17, wherein the programmable graphics
19 rendering pipeline is employed for executing at least one of a partitioned inner product instruction, a
20 partitioned arithmetic instruction, a partitioned logic instruction, a data movement instruction, and a
21 loop-control instruction, to produce at least one of the graphics pixel data and the media output data.

22 22. (Previously Presented) The method of Claim 17, further comprising the step of storing
23 the one of the graphics pixel data and the media output data in memory.

24 23. (Previously Presented) A programmable graphics pipeline for multimedia applications
25 that performs graphics and media functions, comprising:

26 (a) a vector streaming engine that accesses variable length data from a memory
27 and writes pixel data to the memory, wherein the variable length data comprise one of graphic texture
28 source data and media source data, and wherein the pixel data are one of graphics pixel data and
29 media pixel data; and

30 ///

1 (b) a vector processing engine in communication with the vector streaming engine,
2 said vector processing engine generating the pixel data from the variable length data.

3 24. (Original) The programmable graphics pipeline of Claim 23, wherein the vector
4 streaming engine comprises:

5 (a) a vector input unit in communication with the vector processing engine; and

6 (b) a vector output unit in communication with the vector processing engine.

7 25. (Previously Presented) The programmable graphics pipeline of Claim 24, wherein the
8 vector input unit comprises:

9 (a) an enhanced rasterization unit that generates at least one of:

10 (i) a destination coordinate corresponding to one of a graphics pixel
11 location and a media pixel location; and

12 (ii) a source coordinate for one of a graphics texture source, and a media
13 source; and

14 (b) an enhanced texture address unit in communication with the enhanced
15 rasterization unit, said enhanced texture address unit generating a memory address corresponding to
16 the source coordinate, said memory address defining a storage location for one of the graphics texture
17 source data and the media source data; and

18 (c) a texture cache in communication with the enhanced texture address unit, said
19 texture cache storing said one of the graphics texture source data and the media source data for use by
20 the vector processing engine.

21 26. (Original) The programmable graphics pipeline of Claim 24, wherein the vector output
22 unit comprises:

23 (a) a blending unit in communication with the vector processing engine, said
24 blending unit performing blending operations on graphics pixel data;

25 (b) an output buffer in communication with the vector processing engine, said
26 output buffer being employed for concatenating a plurality of media pixel data; and

27 (c) a write buffer in communication with both the blending unit and the output
28 buffer, said write buffer writing one of the graphics pixel data and the media pixel data to memory in
29 burst groups.

30 ///

1 27. (Previously Presented) The programmable graphics pipeline of Claim 23, wherein the
2 vector processing engine comprises:

3 (a) a first vector functional unit in communication with the vector streaming
4 engine, said first vector functional unit being usable for graphics rendering processing functions and
5 media processing functions;

6 (b) an instruction cache in communication with the vector functional unit, said
7 instruction cache storing at least one machine instruction to be executed by the vector functional unit;
8 and

9 (c) a first register file in communication with the first vector functional unit, said
10 first register file storing intermediate data for use by the first vector functional unit.

11 28. (Original) The programmable graphics pipeline of Claim 27, wherein the vector
12 processing engine comprises:

13 (a) a second vector functional unit in communication with the vector streaming
14 engine, and in communication with the instruction cache, said second vector functional unit being
15 usable for performing graphics processing functions and media processing functions in parallel with
16 the first vector functional unit; and

17 (b) a second register file in communication with the second vector functional unit,
18 said second register file storing intermediate data for use by the second vector functional unit.

19 29. (Original) The programmable graphics pipeline of Claim 23, further comprising a
20 configuration register in communication with both the vector streaming engine and the vector
21 processing engine, said configuration register providing a partitioned data size and a location of an
22 instruction indicating operation of the programmable pipeline for one of graphics processing and
23 media processing.

24 30. (Previously Presented) A method for performing one of a graphics function and a media
25 function on variable length data with a programmable graphics pipeline, comprising the steps of:

26 (a) using a vector streaming engine:

27 (i) generating a plurality of output pixel coordinates;

28 (ii) generating a plurality of memory addresses corresponding to the output
29 pixel coordinates at which are stored one of graphics texture data and media source data, wherein the
30 graphics texture data and the media source data are not of equal length; and

- 1 (iii) caching said one of the graphics texture data and the media source data
2 in a cache;
3 (b) passing data retrieved from the cache to a vector processing engine; and
4 (c) performing one of the graphics function and the media function on the data
5 retrieved from the cache using the vector processing engine.
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

1 Examiner's Justification for the Restriction

2 The Examiner's justification for this four-way restriction is that the inventions of the four
3 groups identified by the Examiner are related as subcombinations disclosed as usable together in a
4 single combination, and the subcombinations are distinct from each other because it can be shown
5 that they are separately usable. The Examiner reaches this conclusion by asserting that "invention I
6 has separate utility such as "programmable graphics pipeline;" "invention II has separate utility such
7 as "processing graphics data and media data;" invention III has separate utility such as "processing
8 variable length data;" and "invention IV has separate utility such as "caching texture data and media
9 data." MPEP § 806.05(d) requires that the Examiner provide an indication of the separate utility
10 following the phrase "such as." In his justification of the restriction, the Examiner has simply
11 repeated the phrase he stated to indicate to what each group of claims is drawn, in regard to the
12 different subclasses in which the invention of each group is classified. However, the classification of
13 the claims into four different subclasses as asserted by the Examiner is not justified and therefore, the
14 phrase associated with that incorrect classification cannot be used as a basis for showing that each
15 invention of each group is separately usable.

16 All four groups are indicated by the Examiner as falling under Class 345, which is broadly
17 defined in the Manual of Patent Classification as: "Computer Graphics Processing, Operator
18 Interface Processing, And Selective Visual Display Systems." Applicants agree that the Class for all
19 of the claims in the application is indeed Class 345, but disagrees with the characterization of each
20 group by the Examiner as falling in different subclasses.

21 The Examiner asserts that Claims 1 and 3-16 are classified in Class 345, subclass 506. The
22 manual of patent classification indicates that subclass 506 is indented under subclass 502, which is
23 indented under subclass 501. Subclass 501 broadly covers "A computer graphic processing system:"
24 and more specifically, "subject matter comprising apparatus or method for processing or
25 manipulating data for presentation by a computer prior to use with or in a specific display system."
26 Subclass 502 broadly covers "Plural graphics processors:" and more specifically "subject matter
27 wherein more than one graphics processor is used." Subclass 506 covers "Pipeline processors:" and
28 more specifically, "subject matter wherein the plural processors are operated sequentially." So the
29 Examiner is asserting that the Claims in Group I correspond to a computer graphics processing
30 system having a plurality of graphics processors and more specifically, a plurality of pipeline

1 processors that are operated sequentially. However, applicants' Claim 1 as currently amended in the
2 response to the final Office Action is as follows:

3 A programmable graphics pipeline for processing multimedia data
4 comprising:

5 (a) an instruction cache for storing at least graphics and media
6 instructions;

7 (b) a first register file for storing the multimedia data and
8 intermediate data;

9 (c) a first vector functional unit in communication with the
10 instruction cache and the register file, said vector functional unit performing at least
11 graphics and media instructions, to produce at least graphics and media data; and

12 (d) an enhanced texture cache in communication with the first
13 vector functional unit, the first vector functional unit obtaining from said enhanced
14 texture cache a vector of at least one partition of the multimedia data wherein said
15 enhanced texture cache is not limited to storage of one type of multimedia data.

16 Nothing in Claim 1 refers to a plurality of processors or more specifically, to a plurality of
17 pipeline processors that are operated sequentially. Accordingly, it appears that Claim 1 is not
18 properly classified in subclass 506.

19 The Examiner indicates that the claims in Group II are "drawn to 'processing graphics data
20 and media data' classified in class 345, subclass 546." Subclass 546 is indented under subclass 545,
21 which is indented under subclass 530. Subclass 530 is broadly defined as "Computer Graphics
22 Display Memory System:" and more specifically as "Subject matter wherein a storage system or
23 display memory organization and structure is used for storing image data which is being created and
24 processed for presentation." Subclass 545 is broadly defined as "Frame buffer:" and more
25 specifically defined as "subject matter wherein the graphics display memory stores the contents of a
26 screen of display image." Finally, subclass 546 is broadly defined as "Multi-format frame buffer:"
27 and more specifically as "Subject matter wherein the frame buffer memory stores both video and
28 graphics data, such as, YUV for video and RGB for graphics." Accordingly, a claim properly
29 classified in subclass 546 would have to be drawn to a frame buffer storage system that stores both
30 video and graphics data. However, Claim 17 is as follows:

1 A method for producing one of graphics pixel data and media output data,
2 comprising the steps of:

3 (a) obtaining configuration data from a host processor, said
4 configuration data comprising a partitioned data size and a location of an instruction;

5 (b) performing graphics rendering processing on graphics texture
6 data with a programmable graphics rendering pipeline to produce graphics pixel data
7 when the partitioned data size and the instruction correspond to graphics processing;
8 and

9 (c) performing media processing on media source data with the
10 programmable graphics rendering pipeline to produce media output data when the
11 partitioned data size and the instruction correspond to media processing.

12 There is no mention in Claim 17 of a frame buffer storage system that stores both video and
13 graphics data. Claim 17 recites *producing one* of graphics pixel data and media output data, but does
14 NOT recite any frame buffer that stores two different types of data. Clearly, Claim 17 is not properly
15 classified in subclass 546.

16 The Examiner asserts that the claims in Group III are classified in subclass 522, but broadly
17 pertains to "Graphic command processing:" and more specifically is directed to "subject matter
18 wherein a CPU or a host computer issues a command to a graphic processing system to perform an
19 operation." Claim 23 is as follows:

20 A programmable graphics pipeline for multimedia applications that performs
21 graphics and media functions, comprising:

22 (a) a vector streaming engine that accesses variable length data
23 from a memory and writes pixel data to the memory, wherein the variable length data
24 comprise one of graphic texture source data and media source data, and wherein the
25 pixel data are one of graphics pixel data and media pixel data; and

26 (b) a vector processing engine in communication with the vector
27 streaming engine, said vector processing engine generating the pixel data from the
28 variable length data.

29 Once again, Claim 23 is apparently improperly classified, since there is no mention of a CPU
30 or host processor issuing commands to a graphic processing to perform an operation. Instead, the

1 graphics pipeline is programmable and carries out the functions recited based on the programs with
2 which it is programmed. But there is no indication that a CPU or host processor supplies the
3 commands to make the graphics processor carry out these functions.

4 Claim 30 (Group IV) is classified in subclass 552, which is broadly directed to "Texture
5 memory:" and more specifically, to "subject matter wherein the graphics display memory is used for
6 storing shading and other attribute information where the information is added to the 'surface' of a
7 graphical image or object to mimic the surface detail of real objects." However, Claim 30 is as
8 follows:

9 A method for performing one of a graphics function and a media function on
10 variable length data with a programmable graphics pipeline, comprising the steps of:

- 11 (a) using a vector streaming engine:
 - 12 (i) generating a plurality of output pixel coordinates;
 - 13 (ii) generating a plurality of memory addresses
 - 14 corresponding to the output pixel coordinates at which are stored one of graphics
 - 15 texture data and media source data, wherein the graphics texture data and the media
 - 16 source data are not of equal length; and
 - 17 (iii) caching said one of the graphics texture data and the
 - 18 media source data in a cache;
- 19 (b) passing data retrieved from the cache to a vector processing
- 20 engine; and
- 21 (c) performing one of the graphics function and the media function
- 22 on the data retrieved from the cache using the vector processing engine.

23 It should be apparent that this claim is not specifically directed to storing shading and other
24 attribute information, but is instead is directed to selectively performing one of two different types of
25 functions with a programmable graphics pipeline. Claim 30 also thus appears to be mischaracterized
26 as belonging in subclass 552.

27 It appears that if the Examiner were to determine the proper subclass for each of the
28 independent claims in this application, the claims would be searched in at most, two subclasses.
29 However, since those two groups of claims are generally closely related, very little effort would be
30 required in searching for prior art related to all of the claims in the application, and it would be

1 preferable to search both subclasses anyway, regardless of which group of claims were elected.
2 Indeed, the Examiner has apparently already made such a search. Accordingly, since there has been
3 inadequate justification for issuing this restriction and the reasoning for doing so is not supported, the
4 Director is respectfully requested to require the Examiner to withdraw the restriction and examine all
5 claims in the application.

6 Respectfully submitted,

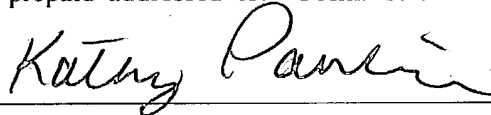
7 
8

9 Ronald M. Anderson
10 Registration No. 28,829

11 RMA/SKM:lrg

12 I hereby certify that this correspondence is being deposited with the U.S. Postal Service in a sealed
13 envelope as first class mail with postage thereon fully prepaid addressed to: Commissioner for Patents,
14 Alexandria, VA 22313-1450, on December 6, 2004.

15 Date: December 6, 2004

16 
17
18
19
20
21
22
23
24
25
26
27
28
29
30